In [51]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import sklearn
```

In [52]:
```
data = pd.read_csv('car-sales-extended-missing-data.csv')
data.head()
```

Out[52]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Honda	White	35431.0	4.0	15323.0
1	BMW	Blue	192714.0	5.0	19943.0
2	Honda	White	84714.0	4.0	28343.0
3	Toyota	White	154365.0	4.0	13434.0
4	Nissan	Blue	181577.0	3.0	14043.0

In [53]:
```
for label, content in data.items():
    if pd.api.types.is_numeric_dtype(content):
        print(label)
```

```
Odometer (KM)
Doors
Price
```

In [54]:
```
for label, content in data.items():
    if pd.api.types.is_numeric_dtype(content):
        if pd.isnull(content).sum():
            print(label)
```

```
Odometer (KM)
Doors
Price
```

In [55]:
```
for label, content in data.items():
    if pd.api.types.is_numeric_dtype(content):
        data[label+'_is_missing'] = pd.isnull(content)
        data[label] = content.fillna(content.median())
```

In [56]:
```
data.isna().sum()
```

Out[56]:
```
Make                        49
Colour                      50
Odometer (KM)                0
Doors                        0
Price                        0
Odometer (KM)_is_missing     0
Doors_is_missing             0
Price_is_missing             0
dtype: int64
```

In [61]:
```
def convert_columns_to_categorical (df, column_list):
    for column in column_list:
        if pd.api.types.is_string_dtype(df[column]):
            df[column] = df[column].astype('category')
        elif pd.api.types.is_numeric_dtype(df[column]):
            df[column] = pd.Categorical(df[column])
        else:
            print(f'Column {column} is not a string or numeric type.')

df = pd.DataFrame(data)
convert_columns_to_categorical(df, ['Make', 'Colour', 'Doors'])
```

In [62]:
```
for column in ['Make', 'Colour', 'Doors']:
    df[column] = df[column].cat.codes

```

In [63]:
```
1 df
```

Out[63]:

	Make	Colour	Odometer (KM)	Doors	Price	Odometer (KM)_is_missing	Doors_is_missing	Price_is_missing
0	2	5	35431.0	1	15323.0	False	False	False
1	1	2	192714.0	2	19943.0	False	False	False
2	2	5	84714.0	1	28343.0	False	False	False
3	4	5	154365.0	1	13434.0	False	False	False
4	3	2	181577.0	0	14043.0	False	False	False
...	...	...	...	...	...	...	...	...
995	4	1	35820.0	1	32042.0	False	False	False
996	0	5	155144.0	0	5716.0	False	False	False
997	3	2	66604.0	1	31570.0	False	False	False
998	2	5	215883.0	1	4001.0	False	False	False
999	4	2	248360.0	1	12732.0	False	False	False

1000 rows × 8 columns

In [64]:
```
1 df.isna().sum()
```

Out[64]:
```
Make                        0
Colour                      0
Odometer (KM)               0
Doors                       0
Price                       0
Odometer (KM)_is_missing    0
Doors_is_missing            0
Price_is_missing            0
dtype: int64
```

In [65]:
```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Make                      1000 non-null   int8
 1   Colour                    1000 non-null   int8
 2   Odometer (KM)             1000 non-null   float64
 3   Doors                     1000 non-null   int8
 4   Price                     1000 non-null   float64
 5   Odometer (KM)_is_missing  1000 non-null   bool
 6   Doors_is_missing          1000 non-null   bool
 7   Price_is_missing          1000 non-null   bool
dtypes: bool(3), float64(2), int8(3)
memory usage: 21.6 KB
```

In [67]:
```
1 df.T
```

Out[67]:

	0	1	2	3	4	5	6	7	8	9	...	990	
Make	2	1	2	4	3	2	4	2	0	2	...	4	
Colour	5	2	5	5	2	4	2	5	5	2	...	5	
Odometer (KM)	35431.0	192714.0	84714.0	154365.0	181577.0	42652.0	163453.0	131821.0	130538.0	51029.0	...	173408.0	235
Doors	1	2	1	1	0	1	1	1	1	1	...	1	
Price	15323.0	19943.0	28343.0	13434.0	14043.0	23883.0	8473.0	20306.0	9374.0	26683.0	...	8082.0	9
Odometer (KM)_is_missing	False	False	False	False	False	False	False	True	False	False	...	False	
Doors_is_missing	False	False	False	False	False	False	False	False	False	False	...	False	
Price_is_missing	False	False	False	False	False	False	False	False	False	False	...	False	

8 rows × 1000 columns

In [71]:
```
from sklearn.ensemble import RandomForestRegressor

%time
model = RandomForestRegressor()
model.fit(df.drop('Price', axis = 1), df['Price'])
```

```
Wall time: 0 ns
```

Out[71]:
```
RandomForestRegressor()
```

In [72]:
```
model.score(df.drop('Price', axis = 1), df['Price'])
```

Out[72]:
```
0.8675889831897092
```

this method can't work cos we only have one dataset which can be used for training and test

In [ ]:
```

```